Blues Bot: Heuristically-Driven Improvisation Over Blues Changes

Bennett Parsons, Jeremy Welborn

Computer Science 182, School of Engineering And Applied Sciences (SEAS)

Algorithmic composition and jazz improvisation

- Improvisation is hardly a well-defined art, not one that a player approaches with a precise plan or policy.
- We are interested in algorithmic composition, leveraging commonly-used, canonical models with our own ideas about good jazz heuristics to build a bot that improvises in the jazz idiom over a given set of chords.

Supervised learning approaches in literature

- Algorithmic approaches to composition have historically relied on Markov models and deep learning, including recurrent neural networks as well as more sophisticated machinery. These supervised techniques have famously been pursued by the Magenta project at Google Brain.
- Our agent speaks Musical Instrument Digital Interface or MIDI, the de facto standard for digitizing music. We found no suitable source of MIDI to start with; we also found in particular that projects tend to build out their own training sets. As such, we were interested in building a data-independent agent.

Our approach: Heuristics

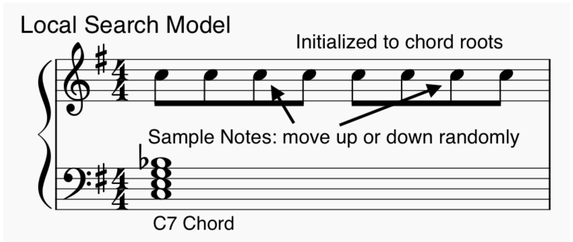
- Can we compose convincing jazz without a training set?
- *Heuristic composition* - Write feature evaluation functions that encourage idiomatic jazz phrases and partial phrases, and string them together to form an complete solo
- Example heuristics for composition - capture *tonal* and *rhythmic* vocabulary, *contour* and *resolution* of phrases

Representation

"Representation is king". This became the crux of our work.

Candidate representations by algorithm:

1. *MDP* – discretize solo into states such that states are only dependent on earlier states. Run an MDP algorithm to compute q-values, and soft max over q-values for a probabilistic policy function. *Advantages:* learned policy could conceivably compose music in real time, good way to leverage feature evaluation functions. *Disadvantages:* no intuitive notion of state, state space too large, hard to build feature evaluation functions that return a reward for a single action (playing one note). ✗
2. *Local Search* – discretize solo into "subproblems": an improvisation over a single chord for one measure. Run a local search algorithm, using feature evaluation functions, to generate a one measure solo, and repeat this process over the entire form. *Advantages:* natural representation, easy to write feature evaluation functions that act over a solo for an entire measure. *Disadvantages:* won't improvise in real time, may not find optimal solution according to our heuristic. But this is actually good, because our heuristics are imperfect! ✓



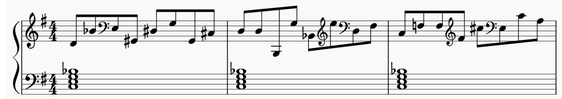
Why local search?

- *Randomness* – comes for free; we want our bot to be constantly exploring new ideas, not finding optimal solutions to our "guiding" heuristics
- *Conditioning* – it's easy to condition on the past as we build a solo measure by measure
- *Speed* – it's fast! $\sim$ 0.4s for 12 bars of solo

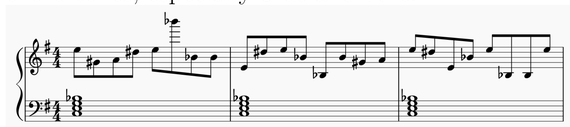
Results and heuristics

We built up our heuristics while relying on implementations of simulated annealing and a genetic algorithm.

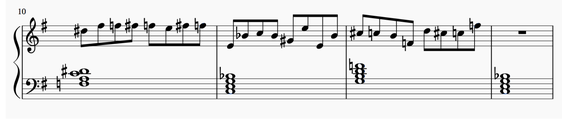
We start with a random stream of 1/8 notes:



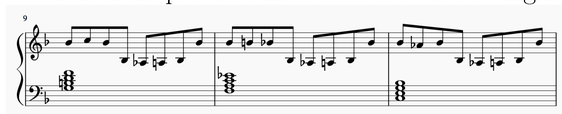
That doesn't sound like jazz! We fix this by adding a feature evaluation function for *tonality*, rewarding chord tones, especially 3rds and 7ths:



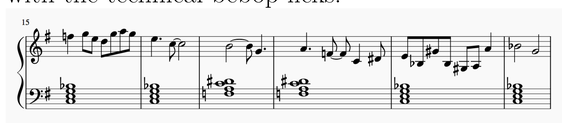
This is better, but there's no shape. We address this by implementing an evaluation function for *contour*, rewarding smooth lines and interval variety. Now the "score" of a solution is the sum of these two functions:



What's missing now is continuity measure to measure. A simple yet especially effective idea addresses this: initialize the "solution" of a measure with the solution of the previous measures. Now it's swingin'!



Finally, we add rhythm with a simple approach: fix the number of notes per measure to be any integer between 1 and 8, then adjust note lengths so that the durations sum to 4 beats. This creates our most convincing output, balancing sparser, lyrical lines with the technical bebop licks.



Discussion: What did we just do?

Our efforts evolved from the straightforward goal of algorithmically generating jazz and jazz lines into a potentially powerful compositional tool and performance companion. With any slight additions, our bot could improvise over any set of changes and produce solos for any jazz tune in existence. The model can extend to other existing genres or be used to create something entirely new – it is capable of making innovative music imitating any specifable heuristics!

Conclusion and future work

- Can always add more and more feature evaluators. Specifically, ones that act across subproblems and encourage motivic coherence throughout the solo
- Extend to other genres. If we can conceive of a reasonable heuristic, then this approach will work.

References and acknowledgments

We wish to thank Professor Kuindersma and our teaching fellow John Chipman for their direction, as well as the apparently influential sounds of:

Contact Information

- GitHub: github.com/bennettparsons/jazz-bot.git
- Bennett: bennettparsons@college.harvard.edu
- Jeremy: jeremywelborn@college.harvard.edu

HARVARD
John A. Paulson
School of Engineering
and Applied Sciences